

# Prompt Bar

**Prompt Bar** The Prompt Bar component is a sophisticated input system designed for modern chat interfaces, AI prompt systems, and messaging applications. It combines a multi-line textarea with customizable action icons, device selection dropdowns, and file attachment capabilities to create a comprehensive input experience. The `<aava-prompt-bar>` component provides an intuitive interface for users to compose messages, select target devices, attach files, and interact with various input actions through a clean, organized layout. How to use ■ `import {`

`AavaPromptBarComponent , PromptIcons , DropdownOption , uploadedImages , }` from

`"@aava/play-core"` ; Explore different prompt bar configurations and features with interactive

demos. **Basic Usage** ■ The Prompt Bar provides a unified interface for user text input, actions, and attachments.

It supports typing, sending messages, and triggering actions via icons.

Use the basic setup to render the default prompt bar with message input and optional send

behavior. **Sizes** ■ **Available Sizes** ■ The Prompt Bar comes in three size variants — `sm`, `md`, and `lg` — allowing flexible adaptation to different layouts.

The size affects the overall height, icon dimensions, textarea rows, and tag scaling.

Use smaller sizes for compact UIs and larger ones for more prominent input sections. `sm` (Small)

– Slightly larger input for standard compact forms `md` (Medium) – Default size for most use cases (balanced proportions) `lg` (Large) – Larger input for prominent forms or accessibility-focused

layouts **Icons & Actions** ■ Prompt Bars can include action icons on both sides of the input field.

Each icon can be interactive — triggering specific actions like attachments, voice input, or emoji selectors.

Icons are customizable with colors, visibility, and click handlers, making it easy to extend the bar's functionality. **File Upload** ■ You can attach files or images to the prompt bar as tags or image previews.

When `showImage` is enabled, supported image formats (`jpg`, `png`, `webp`, etc.) appear as visual thumbnails, while other files display as tags.

This is ideal for chat interfaces or upload workflows. **States & Configuration** ■ The Prompt Bar supports various states such as disabled, and offers configurable options for width, height, placeholder text, and icon layout.

You can also control auto-resizing, selection dropdowns, and behavior for multi-line input.

These settings help integrate the component seamlessly into different UI contexts. **Prompt Bar**

**Features** ■ **Multi-line Input** : Configurable textarea with adjustable rows **Customizable Icons** :

Action icons positioned at start and end slots **Device Selection** : Dropdown for selecting target

devices or platforms **File Attachments** : Visual badges for uploaded images with removal **Size**

**Variants** : Four size options (`xs`, `sm`, `md`, `lg`) for different layouts **Keyboard Shortcuts** : Enter to

send, Shift+Enter for new lines **Responsive Design** : Mobile-optimized layout with touch-friendly

controls **Auto-scroll** : Automatic scrolling to bottom for new content **Features** ■ **Multi-line Text**

Input ■ The core of the prompt bar is a flexible textarea: Configurable Rows : Set the number of visible rows (default: 3) Auto-resize : Textarea adjusts to content while maintaining layout Placeholder Text : Customizable placeholder for user guidance Input Validation : Built-in validation and error handling Accessibility : Full keyboard navigation and screen reader support Icon System ■ Flexible icon positioning and interaction: Slot-based Layout : Icons can be placed in icon-start or icon-end slots Customizable Appearance : Size, color, and visibility controls Click Handlers : Custom click functions for each icon Conditional Display : Show/hide icons based on context Responsive Sizing : Automatic size adjustments based on component size Device Selection ■ Integrated dropdown for target device selection: Custom Options : Define device types with labels and icons Visual Indicators : Icons for each device type Responsive Layout : Adapts to available space Integration Ready : Seamlessly integrates with existing selection systems Accessibility : Full keyboard and screen reader support File Management ■ Visual file attachment system: Image Badges : Display uploaded images as removable tags File Information : Show file size, type, and preview Removal Controls : Easy file removal with click interaction Responsive Layout : Adapts to different screen sizes Visual Feedback : Clear indication of attached files Size Variants ■ Four predefined size configurations: xs (Extra Small) : Compact layout for tight spaces sm (Small) : Standard small layout with balanced proportions md (Medium) : Default size with optimal spacing lg (Large) : Spacious layout for prominent displays API Reference ■ Input Properties ■

Property Type	Default	Description
messages	PromptMessage[]	Array of previous messages for context
placeholder	string	'Type a message' Placeholder text for the input field
disabled	boolean	false Disable the prompt bar input icons
promptIcons	PromptIcons[]	Array of icons to display
rows	number	3 Number of visible rows in the textarea
deviceOptions	DropdownOption[]	Options for device selection dropdown
showSelection	boolean	false Show the device selection dropdown
fileOption	string	" File upload option configuration
size	'lg'   'md'   'sm'   'xs'	'md' Size variant for the prompt bar
uploadedImages	uploadedImages[]	Array of uploaded image attachments
width	number	0 Custom width for the prompt bar
height	number	0 Custom height for the prompt bar
textAreaMaxHeight	number	0 Maximum height for the textarea
textAreaAutoResize	boolean	false Auto-resize textarea based on content

Output Events ■ Event Type Description

messageSent	EventEmitter<string>	Emitted when a message is sent
iconClicked	EventEmitter<{ icon: PromptIcons; currentMessage: string }>	Emitted when an icon is clicked

Interfaces ■ PromptIcons

```

interface PromptIcons {
  name: string; // Icon name/identifier
  click?: () => void; // Optional click handler function
  size?: string; // Icon size (e.g., '20px')
  color?: string; // Icon color (hex, rgb, etc.)
  slot: "icon-start" | "icon-end"; // Position slot
  visible?: boolean; // Whether icon is visible
}

```

DropdownOption ■

```

interface DropdownOption {
  label: string; // Display text for the option
  value: string; // Value associated with the option
  icon: string; // Icon name for the option
}

```

uploadedImages ■

```

interface uploadedImages {
  id: number; // Unique identifier for the image
  label: string; // Display label for the image
  image: string; // Image URL or data
  icon: string; // Icon to display with the image
  size: string; // File size information
}

```

Methods ■ Public Methods ■

```

// Trigger message send programmatically
triggerSend(): void

// Get current message content
getCurrentMessage(): string

// Get icon size based on component size
getIconSize():

```

number // Get icons filtered by slot position getIconsBySlot ( slot : 'icon-start' | 'icon-end' ) :  
PromptIcons [ ] // Get size class for styling getPromptBarSizeClass ( ) : string Event Handlers ■ //  
Handle message sending sendMessage ( ) : void // Handle keyboard input onKeyPress ( event :  
KeyboardEvent ) : void // Handle input changes onInput ( event : Event ) : void // Handle icon  
clicks onIconClick ( icon : PromptIcons ) : void // Handle file badge removal removeBadge ( id :  
number ) : void // Handle device selection changes onSelectionChange ( data : any ) : void CSS  
Custom Properties ■ The prompt bar component uses CSS custom properties for theming:  
Container Properties ■ Property Default Value Description --textbox-background #ffffff  
Background color of the prompt bar --promptbar-border 1px solid #e2e8f0 Border style of the  
prompt bar --promptbar-border-radius 8px Border radius of the prompt bar  
--textbox-textarea-container-padding 16px Padding for the textarea container --textbox-gap 12px  
Gap between elements Textarea Properties ■ Property Default Value Description  
--textbox-textarea-container-padding 16px Padding for the textarea --textbox-gap 12px Gap  
between textarea elements Badge Properties ■ Property Default Value Description  
--badge-default-background #f3f4f6 Background color for badges --promptbar-text-color #6b7280  
Text color for badges Responsive Breakpoints ■ Breakpoint Description max-width: 576px Mobile  
layout adjustments max-width: 360px Small mobile optimizations Best Practices ■ Icon  
Configuration ■ Logical Grouping : Group related icons in the same slot Consistent Sizing : Use  
consistent icon sizes within the same component Meaningful Colors : Use colors that convey  
meaning (e.g., green for success) Accessibility : Ensure icons have proper alt text and  
descriptions Touch Targets : Maintain adequate touch target sizes for mobile Device Selection ■  
Clear Labels : Use descriptive labels for device options Relevant Icons : Choose icons that clearly  
represent each device type Logical Ordering : Arrange options in logical order (e.g., Desktop,  
Tablet, Mobile) Default Selection : Provide sensible default device selection Validation : Validate  
device selection before processing File Management ■ File Types : Support common image  
formats (JPG, PNG, GIF) Size Limits : Implement reasonable file size restrictions Preview Quality :  
Provide clear image previews in badges Removal Confirmation : Consider confirmation for file  
removal Error Handling : Gracefully handle file upload failures User Experience ■ Keyboard  
Shortcuts : Provide intuitive keyboard navigation Visual Feedback : Clear indication of input states  
and actions Responsive Design : Ensure usability across all device sizes Loading States : Show  
loading indicators for async operations Error Messages : Provide helpful error messages and  
recovery options Performance ■ Icon Optimization : Use optimized icon assets Lazy Loading :  
Load non-critical features on demand Event Handling : Efficient event listener management  
Memory Management : Proper cleanup of file references Change Detection : Use OnPush  
strategy for optimal performance

```
<aava-prompt-bar placeholder="Type your message here..." [icons]="allIcons">
</aava-prompt-bar>
```

---

```
onMessageSent(message: string) {
  console.log('Message sent:', message);
}
allIcons: PromptIcons[] = [
  {
    name: 'paperclip',
    slot: 'icon-start',
    color: 'green',
    visible: true,
    click: () => this.onAttachFile()
  },
  {
    name: 'send',
    slot: 'icon-end',
    color: 'var(--color-text-primary)',
    visible: true,
    click: () => this.onAddImage()
  }
];
onAttachFile() {
  console.log('Attach file clicked');
}

onAddImage() {
  console.log('Add image clicked');
}
```

```

<aava-prompt-bar
  size="sm"
  placeholder="Small prompt bar (2 rows)"
  [icons]="allIcons"
>
</aava-prompt-bar>

```

```

<aava-prompt-bar
  size="md"
  placeholder="Medium prompt bar (3 rows)"
  [icons]="allIcons"
>
</aava-prompt-bar>

```

```

<aava-prompt-bar
  size="lg"
  placeholder="Large prompt bar (4 rows)"
  [icons]="allIcons"
>
</aava-prompt-bar>

```

---

```

allIcons: PromptIcons[] = [
  {
    name: 'paperclip',
    slot: 'icon-start',
    color: 'var(--color-text-primary)',
    visible: true,
    click: () => this.onAttachFile()
  },
  {
    name: 'send',
    slot: 'icon-end',
    color: 'var(--color-text-primary)',
    visible: true,
    click: () => this.onSend()
  }
];

```

```

<div>
  <h1>Start Icons Only</h1>
  <aava-prompt-bar
    placeholder="Prompt bar with start icons"
    [icons]="startIcons"
  >
</aava-prompt-bar>
</div>

<div>
  <h1>End Icons Only</h1>
  <aava-prompt-bar placeholder="Prompt bar with end icons" [icons]="endIcons">
</aava-prompt-bar>
</div>

<div>
  <h1>Both Start and End Icons</h1>
  <aava-prompt-bar
    placeholder="Prompt bar with both start and end icons"
    [icons]="allIcons"
  >
</aava-prompt-bar>
</div>

<div>
  <h1>With Select Component</h1>
  <aava-prompt-bar
    placeholder="Prompt bar with select component"
    [icons]="allIcons"
    [showSelection]="true"
    [deviceOptions]="deviceOptions"
    [selectWidth]="'200px'"
    selectPlaceholder="Choose device"
  >
</aava-prompt-bar>
</div>

<div>
  <h1>ng-content with Icons</h1>
  <aava-prompt-bar
    placeholder="Prompt bar with icons and custom content"
    [icons]="allIcons"
  >
    <div class="custom-content-with-icons">
      <button class="action-button" (click)="onCustomAction()">
        Custom Action
      </button>
      <span class="status-text">Ready</span>
    </div>
  </aava-prompt-bar>
</div>

<div>
  <h1>ng-content Only (No Icons)</h1>
  <aava-prompt-bar placeholder="Prompt bar with only custom content">
    <div class="custom-content-only">

```

```

        <button class="primary-button" (click)="onSaveDraft()">Save Draft</button>
        <button class="secondary-button" (click)="onClear()">Clear</button>
        <button class="primary-button" (click)="onSendMessage()">
            Send Message
        </button>
    </div>
</aava-prompt-bar>
</div>

```

```

<div>
    <h1>Custom Third Row Content (ng-content)</h1>
    <aava-prompt-bar placeholder="Prompt bar with custom third row content">
        <div class="custom-third-row">
            <div class="custom-left">
                <button class="custom-button">Custom Action</button>
            </div>
            <div class="custom-center">
                <span class="custom-text">Custom Content</span>
            </div>
            <div class="custom-right">
                <button class="custom-button">Send</button>
            </div>
        </div>
    </aava-prompt-bar>
</div>

```

---

```

startIcons: PromptIcons[] = [
    {
        name: 'paperclip',
        slot: 'icon-start',
        color: 'var(--color-text-primary)',
        visible: true,
        click: () => this.onAttachFile()
    },
    {
        name: 'image',
        slot: 'icon-start',
        color: 'var(--color-text-primary)',
        visible: true,
        click: () => this.onAddImage()
    }
];

```

```

endIcons: PromptIcons[] = [
    {
        name: 'mic',
        slot: 'icon-end',
        color: 'var(--color-text-primary)',
        visible: true,
        click: () => this.onVoiceRecord()
    },
    {
        name: 'send',

```

```

        slot: 'icon-end',
        color: 'var(--color-text-primary)',
        visible: true,
        click: () => this.onSend()
    }
];

allIcons: PromptIcons[] = [
    ...this.startIcons,
    ...this.endIcons
];

deviceOptions: PromptBarOption[] = [
    { label: 'Desktop', value: 'desktop', icon: 'monitor' },
    { label: 'Tablet', value: 'tablet', icon: 'tablet' },
    { label: 'Mobile', value: 'mobile', icon: 'smartphone' }
];

onMessageSent(message: string) {
    console.log('Message sent:', message);
}

onIconClick(event: any) {
    console.log('Icon clicked:', event);
}

onAttachFile() {
    console.log('Attach file clicked');
}

onAddImage() {
    console.log('Add image clicked');
}

onVoiceRecord() {
    console.log('Voice record clicked');
}

onSend() {
    console.log('Send clicked');
}

onCustomAction() {
    console.log('Custom action clicked');
}

onSaveDraft() {
    console.log('Save draft clicked');
}

onClear() {
    console.log('Clear clicked');
}

onSendMessage() {
    console.log('Send message clicked');
}

```



}

■ No code found

■ No code found